AMENDMENT BY SUBSTITUTE SPECIFICATION
35 C.F.R. §1.121(b)(3) and §1.125
Marked Version

TRANSCRIPT MANAGEMENT SOFTWARE AND METHODS THEREFOR

CROSS-REFERENCE TO RELATED APPLICATION

This application claims priority under 35 U.S.C. 119(e) from U.S. provisional patent application Serial No. 60/221,804, entitled "TRANSCRIPT MANAGEMENT SOFTWARE AND METHODS THEREFOR," filed July 31, 2000, the disclosure of which is hereby expressly incorporated by reference herein.

FIELD OF THE INVENTION

This application relates, in general, to transcript management software and methods therefore.

BACKGROUND OF THE INVENTION

Legal transcripts are used to memorialize the events which occur in a deposition or courtroom proceeding or the like. After a transcript has been taken of a deposition or court proceeding, a court reporter provides a version of the transcript, either a draft version or a final version, to attorneys, courts, witnesses, or other parties interested in the transcript. A draft or "dirty" or non-final version of a transcript typically has words which are later changed or corrected by the court reporter or by the witness prior to the issuance of a final, clean version of the transcript. Spelling errors, abbreviations, phonetic spellings, are a few examples of the items

which are corrected prior to issuance of a final clean transcript. Conventionally, the attorney or paralegal reviews the transcript, such as the draft transcript, and highlights text within the transcript of interest, or makes notes regarding portions of text within the transcript.

As recognized by the present inventors, with the advent of electronic transcripts, there is a need for transferring electronic annotations that are associated with a draft of a transcript so that the annotations are properly associated with a final version of the transcript, so that the annotations can be readily used when referencing the final version of the transcript. Further as recognized by the present inventors, there is a need for analyzing the structure of a selected transcript in order to automatically identify the different sections within a transcript, such as the question and answer pairs, a change of witnesses, a direct examination, a cross-examination, reference to an exhibit, etc.

It is against this background that the various embodiments of the present invention were developed.

SUMMARY OF THE INVENTION

In light of the above and according to one broad aspect of one embodiment of the invention, disclosed herein is a method for transferring annotations associated with a draft electronic transcript to a revised electronic transcript. The method determines if the revised electronic transcript is derived from the draft electronic transcript, and if so, creates annotations in the revised electronic transcript based on the annotations associated with the draft electronic transcript.

The foregoing and other features, utilities and advantages of the invention will be apparent from the following more particular description of various embodiments of the invention as illustrated in the accompanying drawings and claims.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 illustrates a block diagram in accordance with one embodiment of the present invention.

Fig. 2 illustrates an example of a screen display in accordance with one embodiment to the present invention.

Fig. 3 illustrates the general logical operations of the transcript management software in accordance with one embodiment of the present invention.

Fig. 4 illustrates an example of a screen display showing an annotation in the transcript pane of the display relating to the "default issue" in accordance with one embodiment of the present invention.

Fig. 5 illustrates an example of a screen display with a comment which can be associated with a particular annotation in accordance with one embodiment of the present invention.

Fig. 6 illustrates an example of a screen display wherein the display pane has annotations, issue codes, and comments associated therewith.

Fig. 7 illustrates an example of a screen display with a quick preview pane associated with the word index, in accordance with one embodiment of the present invention.

Fig. 8 illustrates an example of a screen display with a pull down search menu, in accordance with one embodiment of the present invention.

Fig. 9 illustrates an example of a screen display showing the structural elements of a particular transcript within the display pane in accordance with one embodiment of the present invention.

Fig. 10 illustrates the logical operations for importing a transcript into the transcript management software, in accordance with one embodiment of the present invention.

Fig. 11 illustrates an example of a screen display showing a real time transcript in the transcript pane in accordance with one embodiment of the present invention.

Fig. 12 illustrates the logical operations for processing real time transcript data, in accordance with one embodiment of the present invention.

Fig. 13 illustrates the logical operations for synchronizing a final transcript to an earlier draft transcript in accordance with one embodiment of the present invention.

Figs. 14 A-B illustrate the logical operations for linking the lines and words of a final transcript to a draft transcript, in accordance with one embodiment of the present invention.

Fig. 15 illustrates an example structure of a line list of a transcript and a word list of a transcript in accordance with one embodiment of the present invention.

Fig. 16 illustrates the logical operations for linking the line and word list of a final transcript to the line and word lists of a draft transcript, in accordance with one embodiment of the present invention.

Figs. 17A-D illustrate examples of line and word lists of a draft and final transcript as they are linked using the operations of Fig. 16, in accordance with one embodiment of the present invention.

Figs. 18-19 illustrates the logical operations for relocating an annotation from a draft transcript and to a final transcript, in accordance with one embodiment of the present invention.

Figs. 20A-D illustrate an example of word lists for a draft and final transcript having links there between, and how the operations of Figs. 18-19 are performed thereon, in accordance with one embodiment of the present invention.

Fig. 21 illustrates the logical operations for analyzing the structural components of transcript data from a real time stream or a transcript file, in accordance with one embodiment of the present invention.

Figs. 22A-B illustrate an example of transcript data analyzed in accordance with one embodiment of the present invention.

Figs. 22A-B illustrate an example of analyzing transcript data into lexical units in accordance with one embodiment of the present invention.

Fig. 23 illustrates an example of transcript data to be analyzed in accordance with one embodiment of the present invention.

Fig. 24 illustrates one example of logical operations for analyzing the structural components of a transcript, in accordance with one embodiment of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

In accordance with the various embodiments of the present invention, a method and computer program product for organizing, annotating, reporting, and managing electronic transcripts is disclosed herein. In particular, methods for automatically analyzing the structural components of an electronic transcript are disclosed herein, as well as methods for transferring annotations associated with a draft electronic transcript to a revised or final electronic transcript for use therewith is also disclosed herein. Other features and functions of various embodiments of the present invention are also disclosed herein.

Fig. 1 illustrates a block diagram of one embodiment of the present invention. As shown in Fig. 1, a court reporter 10 records the events and words spoken during a deposition, court proceeding, or the like, using, for example, a CAT (computer-aided transcription) system 11 coupled to a computer 12. The resulting transcript of the proceeding is stored in a file 13 which can be persistently stored on a removable media such as a floppy disk, CDRom, or the like, or transferred over a network or a serial link to a user such as a court, a judge, a lawyer, a law firm, a witness, etc. In one example, the transcript file is stored in a transcript database 15 having a server 16 coupled thereto, which makes the electronic transcript available for access by others over a network 14.

In accordance with embodiments of the present invention, one or more user computing stations 17 are provided with transcript management software 18 which permits multiple users to access one or more electronic transcripts. The transcript management software 18 permits a user to organize, annotate, search, and manage a plurality of transcripts. Multiple users can access a given transcript and perform various functions on a transcript, in that the transcript is a document sharable by multiple users.

In one example, the electronic transcript generated by the court reporter 10 and accessible by one or more users 17 in the computing system is a draft or final transcript. A draft or "dirty" transcript generally is a non-final version of the transcript and can be a complete transcript stored as a complete file, or alternatively, can be comprised of real time data being received by one or more users having the transcript management software 18 of the present invention.

A final or "clean" transcript is the final version of the transcript. As used herein, the term "final" transcript includes a final transcript or a non-final transcript which has been revised since the draft transcript was provided to the user. Preferably, the final transcript is embodied in a file

which has been electronically signed by the court reporter and can be authenticated by its recipients as, in one example, is described in co-pending U.S. Patent Application Serial No. 60/209,809 filed June 5, 2000, entitled APPARATUS, SYSTEM, AND METHOD FOR ELECTRONICALLY SIGNING ELECTRONIC TRANSCRIPTS, which is incorporated by reference herein in its entirety. Also incorporated herein by reference is U.S. Provisional Patent Application Serial No. 60/212,654 filed June 19, 2000, entitled DYNAMIC SEARCH-DRIVEN WORD INDEX FOR AN ELECTRONIC TRANSCRIPT DISPLAYABLE ON HAND-HELD PORTABLE COMPUTING DEVICE.

In one example, the transcript file (either draft or final) can be received via e-mail, a floppy disk, CDRom, network connection, serial port, or the like. Further, the transcript file can be an ASCII file with or without page numbers and line numbers, being in AMICUS format, a page image format, an electronic transcript format as an executable, or in a PTX format, a live note PTF file, an XML transcript, or a transcript resulting from a real time feed from a serial line (i.e., RS232), the internet, or a file, or the like.

Referring to Fig. 2, an example of a display screen of the transcript management software 18 is shown. Referring to Fig. 2, a transcript pane 21 contains the text of the electronic transcript being presently displayed to the user. The transcript pane 21 includes a scroll bar 22, a page indicator 23, and the title 24 of the transcript.

A display pane 25 displays various information depending upon what the user has selected within the tree control pane. For example, the display pane 25 can show a list of transcripts 27 which form a user-selected group of transcripts (see Fig. 2), a list of issue codes or annotation subjects 28 (see Fig. 4), a list of issues and comments 29 as they appear in one or

more transcripts (see Fig. 6), a table of contents 30 relating to the structural elements of a particular transcript (see Figs. 7, 8, and 9).

The tree control pane 26 generally illustrates a tree structure of a project, wherein a project is a collection of one or more transcripts, and the collection of transcripts can be subdivided into subgroups. In accordance with embodiments of the present invention, a variety of operations can be performed on the transcripts within a project, such a search operation, a report generation operation which creates a search report or an annotation report, an annotation operation, an issue code operation, a comment operation, a table of contents with hyperlinks, and a printing operation, for example. The tree control pane 26 shows the various subcomponents of a project which include annotations 31, reports 32, issues 33, search groups 34, and transcripts 35, being either static transcripts or live transcripts received from a real time feed (see Fig. 11 for example).

An issue pane 36 is shown which lists the name 37 of an issue (such as "negligence," "omission," "default issue") as well as a user-selected highlighting color associated with the particular issue. Further, as shown in Fig. 4, upon the user selecting "All Issues" 38 from the tree control pane 20, the display pane 25 shows the master list 39 of all issues available for the project, and permits the user to enable or disable the display of particular issues within the issue pane 36. For instance, if the user selects the issue labeled "default issue" in the issue pane 36, then when the user makes a new annotation, the new annotation is associated with the "default issue" and highlighted within the transcript in the appropriate color.

Generally, an annotation includes one or more issues or issue codes. The annotation/issue code can include a comment field wherein the user can further add a descriptive comment associated with a particular annotation occurring at a particular location of the

transcript, as shown in Figs. 5-6. The process for making an annotation to a particular section of an electronic transcript is as follows. The user activates the creation of a new issue, and then supplies the name for the issue and selects a corresponding highlighting color. In one example, a plurality of colors is provided from which the user can select to associate the particular issue with a particular highlighting color. The software then adds the new issue to the master list 39 of all issues, and the user can then select text within a transcript to highlight and tag or associate with the particular annotation/new issue. In this manner, the selected text is associated with the selected issue and code. The annotation is associated with the transcript file and can be viewed and accessed by others who later access the transcript file, in one example of the present invention.

Referring again to Fig. 2, a word index pane 40 contains a display of a hyperlink word index for all of the words occurring in a transcript, and preferably for all of the transcripts in a project. In one example, the scope of the word index can be limited to subgroups of transcripts selected by the user. A quick search drop-down menu 41 is provided wherein the user can type in words to initiate a quick search. Alternatively, a full Boolean search can be performed as well. Preferably, the quick search menu 41 has a drop-down feature 42 which remembers the last set of searches entered by the user, as shown in Fig. 8. Furthermore, the word index pane 40 can display a quick preview of the transcript which shows a word selected by the user in context within a smaller preview display of the text around the word selected by the user. For instance, as shown in Fig. 7, the word index pane 40 includes the word "ab" which the quick preview pane 43 shows in context with the words of the transcript occurring around the word "ab" displayed therein. This permits the user to preview various occurrences of words in the word index pane prior to jumping to those words within the display of the transcript pane.

Referring now to Fig. 3, the general operations of the transcript management software are illustrated. Referring to Fig. 3, at operation [[1]]44, the user creates or opens a project within the transcript management software 18. As previously mentioned, a project can include one or more transcripts which can be organized into groups or subgroups, as well as reports, search groups, annotation groups, or the like. At operation [[2]]45, the user imports a transcript into the project. In one example, the transcript imported is a transcript which is a "draft" transcript, a "final" transcript, or a transcript resulting from a real time datastream. As will be explained below and with reference to Figs. 10-12, the transcript management software 18 performs various operations in order to import a transcript into the transcript management software 18. These operations include analyzing the contents of a transcript to detect structural elements such as the beginning of a direct examination, a cross-examination, question and answer pairs, a change of speaker, or the like (Figs. 10, 21-23); processing data from a real time transcript (Fig. 12); and transferring the annotations associated with a draft transcript to a final transcript for use therewith (Figs. 13-20).

Again referring to Fig. 3, at operation [[3]]46, the user creates and defines annotation and issue codes to be associated with the transcript presently being operated upon by the user. Further, the user can view the various annotations and issue codes present within the transcript, as well as perform other functions as previously described herein, such as quick searching, creation of a report including search results, printing, etc., as shown at operation [[4]]47 of Fig. 3.

Referring now to Fig. 10, one example of the logical operations for importing a transcript in accordance with one embodiment of the present invention is illustrated. At operation [[1]]48, the user selects the transcript file for import. The user can associate metadata with the file at

operation [[2]]49, for example, the name of the proceeding, the volume number of the transcript, the deponent's name, the parties, whether the transcript is final or draft, the date, time, type of transcript (deposition of a witness or an expert), and the like. At operation [[3]]50, the user selects the project destination by selecting an appropriate folder in the tree control. At operation [[4]]51, the software 18 analyzes the content of the electronic transcript to detect its structural elements. Preferably, the software 18 reads the page number and line numbers of the transcript, and detects witnesses, change of witnesses, question and answer pairs, direct examination, cross-examination, each new quotation, and references to exhibits. Figs. 21-23 and the text associated therewith describe operation [[4]]51 of Fig. 10 in greater detail in accordance with one embodiment of the present invention. At operation [[5]]52 of Fig. 10, a hyperlinked table of contents is created based upon the structural elements detected at operation [[4]]51. The user can edit the hyperlinked table of contents if desired at operation [[6]]53. As shown in Fig. 9, the table of contents 30 of the transcript can identify sections of a transcript corresponding to witness testimony, direct examination, change of witnesses, proceedings, exhibit index, a general index, and appearances, or the like.

At operation [[7]]54, a word index of the transcript is created for display within the word index pane 40 of the display 20. Preferably, the word index is integrated with the word index of other transcripts within a project so that the word index is a "global" word index having references to words appearing in one or more transcripts of a project. At operation [[8]]55, preferably the transcript, metadata, hyperlinked table of contents, and word index are written to a database for the project. At this point, the transcript is then usable and accessible to the user within the transcript management software.

The operations as illustrated in Fig. 10 are generally applicable for importing a transcript file. Embodiments of the present invention further support the importation of real time transcript data as illustrated in Fig. 11 and by the logical operations of Fig. 12. Referring to Fig. 12, at operation [[1]]56, the user selects the import type and settings, such as a serial import (which preferably includes a baud rate setting), an internet import (optionally with a user name and password), or a file import which contains real time transcript data therein, and the like. At operation [[2]]57, the user can associate metadata with the real time transcript data, in a similar manner as described with reference to operation [[2]]49 of Fig. 10. At operation [[3]]58, the metadata is stored to the database, and at operation [[4]]59, the location of the repository for the raw datastream of the real time transcript data is specified so that the real time transcript data can be stored as it is received by the transcript management software. At operation [[5]]60, a streaming state is initiated wherein real time transcript data is received by the transcript management software and processed therein. As the data is streaming, the transcript management software 18 permits the transcript pane 21 of the display to be either a streaming display, or a snapshot display. Preferably, the snapshot display is a display which is paused and can be manipulated by the user within the transcript pane 21 of the display, while in a background process the streaming data is still being received and stored in an internal buffer of the transcript management software 18. Preferably, the snapshot display mode is enabled or disabled through the use of a single keystroke operation, such as the depression of the return key, or the like. Further, during the streaming state operations, the transcript management software 18 analyzes the content of the received transcript data, in a manner similar to operation [[4]]51 of Fig. 10. Further, a word index is dynamically created as the streaming data is received.

As real time streaming transcript data is received, in accordance with the present invention, words and phrases of the streamed data can be dynamically marked through an "auto annotation" feature of the present invention. The auto annotation operations include providing for a user to specify one or more words or phrases which should be highlighted using a particular issue code which preferably includes a highlight color and an issue description. As the streaming text is received by the transcript management software 18, whenever the particular word or phrase specified by the user is received, the software 18 automatically creates an annotation marking the specified word or phrase using the specified issue code. The user can specify the desired words or phrases for annotation either prior to entering the streaming state, or during receipt of streaming data (such as within the snapshot display mode).

Further, dynamic annotations using a quick annotation feature (which preferably applies a one-line annotation to the current streaming line received during a real-time session, or issue codes, described ~~above,~~ above) can also be made during the streaming state, preferably during a snapshot display of the transcript text. Preferably, annotations are stored to the database as they are created. Because the streaming data is analyzed as it is received within the streaming state, embodiments of the present invention permit dynamic searching, report generation, and printing of transcript data as it is received. At decision operation [[6]]61, if the stream is complete, then control is passed to operation [[7]]62 wherein the raw data stream are written as a draft or "dirty" transcript file into a database for persistent storage.

As illustrated in Fig. 12, storage of real time transcript data results in a draft transcript. The court reporter would, at some time later, issue a final or revised transcript which corrects any typographical errors and preferably contains a certification page and an electronic signature within the final transcript as described in the "APPARATUS, SYSTEM, AND METHOD FOR

ELECTRONICALLY SIGNING ELECTRONIC TRANSCRIPTS" patent application, referenced above. In accordance with the present invention, the annotations created and associated with a draft transcript can be transferred to and associated with a final electronic transcript for use therewith. While a draft transcript was described with reference to Fig. 12 as being a result of receiving real time transcript data, it is understood that a draft transcript can include more generally any non-final draft version of the transcript, either from a real time datastream or from an e-mail transcript file sent by the court reporter soon after the deposition or court proceeding being transcribed was finished.

Figs. 13-20 illustrate examples of the logical operations for creating annotations in a final transcript based on the annotations previously created in a draft transcript file. These operations assume the existence of a draft electronic transcript file which the user has created and associated therewith annotations and issue codes as previously described. Further, these operations assume that the court reporter has provided a final or revised electronic transcript file which is accessible to the user through the transcript management software. Referring to Fig. 13, at operation [[1]]63, the user selects the draft transcript file in a project, and at operation [[2]]64, the user selects the final transcript file for import. Preferably, the operations for importing a transcript file shown in Fig. 10 are performed, with the exception that operations [[7-8]]54-55 of Fig. 10 (writing data to the database) are preferably not performed until later, as will be described below.

At operation [[3]]65 of Fig. 13, a matching operation is performed which confirms that the final transcript file was derived from the draft transcript file. In one example, the final and draft transcript files are analyzed using a match algorithm to confirm that the final file appears to be derived from the draft file. In one example, operation [[3]]65 of Fig. 13 can be implemented as shown in Figs. 14-17. After confirming that the final transcript was derived from the draft

transcript, operation [[4]]66 of Fig. 13 creates annotations in the final transcript file based on the annotations associated with the draft transcript file. In one example, operation [[4]]66 of Fig. 13 can be implemented as shown in Figs. 18-20. Operation [[5]]67 of Fig. 13 then writes the final transcript, the metadata associated therewith, the hyperlinked table of contents, and the word index to the database. Preferably, operation [[5]]67 replaces the draft transcript with the final transcript and the annotations created by the user in the draft transcript file are automatically moved or associated with the final transcript file.

Fig. 14A illustrates one example of the logical operations for performing a matching operation to confirm that the final transcript file was derived from the draft transcript file, in accordance with one embodiment of the present invention. For both the final and draft transcript files, preferably a line list and a word list is created for each file, and the contents of the lists are compared and linked to determine if the final transcript file was derived from the draft transcript file and to determine where the annotations from the draft transcript should be located within the final transcript file. Operations [[1-4]]68-71 of Fig. 14A generate the word lists and line lists for the draft and final transcripts, and it is understood that the order in which operations [[1-4]]68-71 are performed is a matter of choice depending on the particular implementation.

At operation [[1]]68, a "word list" for the draft transcript is formed which preferably has, as entries in the list, a hash of each word of the transcripts. A hash operation is performed on each word of the draft transcript and a list is created thereof. The length of the word list is preferably the number of words in the draft transcript.

In one example, the hash operation is preferably as shown in Fig. 14B. A hash operation in effect creates a unique numerical representation of a string of characters. Preferably, as shown in Fig. 14B, the hash algorithm ignores white spaces and, for a given unique string input into the

hash algorithm, a unique number is preferably returned. In this way, the content of a particular string (such as a word or a line of words) of a transcript can be uniquely represented with a single numerical value. Ideally, if the hashed numerical representation of a word from the final transcript is identical to the hashed numerical representation of a word from the draft transcript, then the words of the transcripts represented by the hashed values are identical.

Preferably, each entry in the word list for the draft transcript also has a page number and a line number indicating where the word being hashed occurred within the draft transcript, a start position indicating the character position within the line where the word started within the draft transcript (i.e., the number of spaces measured from the left-most side of the transcript to the first character of the word), and the end position of the word (i.e., the number of character spaces measured from the right most side of the transcript to the last character of the word).

At operation [[2]]69 of Fig. 14A, a "line list" for the draft transcript is created, wherein each entry in the list includes a hash of the line of the draft transcript, and the page number and line number of where the line being hashed occurred within the draft transcript. At operation [[2]]69, a hash operation is performed on each line of the draft transcript, and preferably, the list created therefor has an entry for each numerical value resulting from the hash operation performed on each line of the draft transcript, the numerical values representing the content of each line of the draft transcript. The hash operation preferably is as shown in Fig. 14B. In one example, the number of entries in the line list is equal to the number of lines in the draft transcript.

Together operations [[1]]68 and [[2]]69 create lists which are representative of the contents of the draft transcript. Fig. 15 illustrates one example of the structure of a line list 72 for a transcript, and a corresponding word list 73 for the transcript in accordance with one

embodiment of the present invention. As shown in Fig. 15, the line list 72 is shown as having a plurality of entries 74, preferably in sequential order; while the word list 73 for the transcript is shown as having the plurality of entries 75, preferably in sequential order, wherein each entry has a corresponding link or pointer 76 to the respective line at which the word occurs within the given transcript, and vice versa.

Referring back now to Fig. 14A, at operation [[3]]70, word list for the final transcript is formed in a manner similar to operation [[1]]68 of Fig. 14A, in that each entry in the word list for the final transcript contains a hash of each word of the final transcript, and the page/line number and start/end position of the word as it occurred in the final transcript.

At operation [[4]]71, a line list for the final transcript is formed in a manner similar to operation [[2]]69 of Fig. 14A, in that the entries of the line list each have a hash of the line from the final transcript and the page/line number of where the line being hashed occurred within the final transcript. Accordingly, the line list 72 and word list 73 for the final transcript will generally have the structure as shown in Fig. 15.

In general at operation [[5]]76 of the Fig. 14A, the transcript management software 18 matches the lines between the draft and final transcript by using the line lists and comparing the hashes contained within each entry of the line lists. For each matching line as determined by operation [[5]]76, the word list entries associated with those matching lines are also matched together by comparing the hashes of the words contained within the respective entries of the word lists.

At operation [[6]]77, the transcript management software 18 determines if the final transcript appears to be derived from the draft transcript, and if so, annotations from the draft

transcript can be associated with the final transcript in the proper respective positions, as will be described below.

Fig. 16 illustrates one example of the logical operations for matching the lines and words of the line lists and word lists of the final transcript and the draft transcript, in accordance with one embodiment of the present invention.

The operations in Fig. 16 will be described, and examples of these operations will be shown with reference to Figs. 17A-D for purposes of illustration. Fig. 17A illustrates an example line list <u>78</u> for a draft transcript and an example of a line list <u>79</u> for a final transcript. The example draft line list <u>78</u> has seven entries with the hash value indicated within each entry. For instance, the second entry <u>80</u> in the example line list <u>75</u> for the draft transcript shown in Fig. 17A has a hash value of five for illustrative purposes. Fig. 17A also shows an example of a line list <u>79</u> having entries representing the contents of a final transcript, wherein each entry is shown having the hash value of the line of the final transcript, for illustrative purposes. Figs. 17B-D illustrate one example of the operations of Fig. 16, as will now be described.

Referring to Fig. 16, operation [[1]]<u>81</u> defines a section that preferably encompasses or covers the entries in the draft line list <u>78</u> and the final line list <u>79</u>. One example of operation [[1]]<u>81</u> is shown in Fig. 17B, wherein a section S_0 has been defined to cover the entries in both the line list of the draft transcript and the line list of the final transcript.

Referring to Fig. 16, operation [[2]]<u>82</u> searches for unique matching line entries between the line lists for the draft and final transcripts within the section defined by operation [[1]]<u>81</u>. In one example, each entry in the draft line list <u>78</u> is examined and the hash value thereof is compared to the hash value of each entry in the final line list <u>79</u> to determine if there is a unique match therebetween. In this sense, a match is "unique" preferably if the hash value matched

between entries occurs only once in each respective defined section of both line lists. In other words, if an entry the draft line list 78 has a identical matching hash value which occur in more than one of the entries in the final line list 79, then the match is not unique. Conversely, if an entry in the draft line list 78 has a hash value which occurs identically in only one entry of the defined section of the final line list 79, then the match is unique.

Operation [[3]]83 of Fig. 16 creates a link 84 from the draft line list 78 to the final line list 79 for each pair of uniquely matching entries. In one example, the link is formed by providing, within each entry of the uniquely matching pair, an address or pointer to the other entry of the uniquely matching pair. Referring now to Fig. 17B, examples of operations [[2]]82 and [[3]]83 are illustrated. For instance, referring to the top entry 85 in the draft line list 78, the hash value therein is two, which uniquely matches the top entry 86 in the final line list 79 because no other entries in the final line list 79 have a hash value of two. Accordingly, a link 87 is formed therebetween. Similarly, the entry 88 in the draft line list 78 having a hash value of one uniquely matches an entry 89 in the final line list 79, and accordingly a link 90 is made therebetween. Note that a link is not made at this time between any other entries because the remaining entries, while some are matching entries, are not uniquely matching entries since their respective hash values occur more than once.

Referring again to Fig. 16, operation [[4]]91 determines if the first and last set of entries within the defined section are matching entries, regardless of their uniqueness and a link is formed therebetween if the entries have matching hash values. An example of this operation is shown in Fig. 17B wherein the bottom entries 92,93 both have the hash value of 7, and therefore a link 94 is formed therebetween, despite the fact that the value 7 occurs more than once within the final line list 79.

Operation [[5]]95 of Fig. 16 attempts to link the entries above and below any previously created link regardless of uniqueness. Preferably, if the adjacent set of entries above is already linked, then operation [[5]]95 attempts to link the set of entries adjacent below the present link. An example of operation [[5]]95 is illustrated in Fig. 17B, wherein beginning with the second link 90 between the third entries having hash values of 1, the adjacent pair of entries below are matching entries 88,89 and accordingly a link 96 is made therebetween by operation [[5]]95. Similarly, for the last set of entries 92,93 having a hash value of 7, the adjacent set of entries 97,98 above are matching entries and a link 99 is created therebetween by operation [[5]]95, in one example.

Referring again to Fig. 16, operation [[6]]100 then eliminates the sections previously defined in both the draft and final line lists. Operation [[7]]101 then recalculates the sections in order to include a plurality of sections wherein a section is formed for each consecutively linked set of entries, or a section is formed for each consecutively non-linked set of entries. One example of operations [[6-7]]100-101 is shown in Fig. 17C wherein a section S_1 is created between the first pair of linked entries 102,103, a section S_2 is created for the next pair of non-linked entries 104,105, a section S_3 is created to include the next two pairs of linked entries 106,107 and 108,109 since they are consecutive, a section S_4 is created to include the next set of non-linked entries 110,111 (shown as entries having a hash value of 7, and assumed at this time to be non-linked as shown in Fig. 17B), and a section S_5 to include the final two pairs of linked entries 112,113 and 114,115.

Operation [[8]]116 of Fig. 16 performs operations [[2-5]]82, 83, 91 and 95 on each non-linked section only, wherein a match is considered unique when evaluated within a particular section as created by operation [[7]]101. One example of operation [[8]]116 is illustrated into

Fig. 17C, wherein the non-linked section includes section S_2 and section S_4. A link cannot be made in section S_2 since the hash values of the entries 104,105 between the line list do not match in section S_2. In section S_4, the entries have matching hash values (shown as 7) and these hash values are unique within section S_4, therefore operation [[8]]116 creates a link 117 therebetween. Since there are no more non-linked sections, operation [[8]]116 is complete. Operation [[9]]118 of Fig. 16 loops to operation [[6]]116 until no further links are made within any non-linked sections.

Operation [[10]]119 then links corresponding entries from the word lists of both the draft and final transcripts based on any link made between entries of the respective line list. In one example, operation [[10]]119 can be performed as each link is made. Because a link between lines indicates equality therebetween, the words of these lines will also be linked. Operation [[11]]120 performs operations [[1-9]]81, 82, 83, 91, 95, 100, 101, 116 and 118 of Fig. 16 as applied to the draft word list 78 and the final word list 79 in order to define sections within the word lists, and match entries therein. One example of operation [[10-11]]119,120 of Fig. 16 is illustrated in Fig. 17D, wherein the word list entries corresponding to the first set of linked line lists entries are matched to the extent that the word list entries have identical hash values and meet the criteria of operation [[11]]120 of Fig. 16 (which includes the criteria of operations [[1-9]]81, 82, 83, 91, 95, 100, 101, 116 and 118 of Fig. 16).

Having completed the links between the line entries and the word entries of the lists for the draft and final transcripts, operation [[12]]121 queries the user as to whether the draft and final transcript files are in fact related based upon one or more metrics. Preferably, if the percentage of linked entries between the draft and final word lists is less than 60%, then the user is queried as to whether the draft and final transcripts are in fact related files. It is understood

that the particular percentage criteria used in operation [[12]]121 is a matter of choice dependent upon the particular implementation.

The following pseudo-code illustrates one embodiment of the logical operations of Fig. 16:

definitions for this example
List entry: A hash that corresponds to a line/word in a hash list.
Section: Consecutive list entries that are either linked or unlinked.
Linked entry: a line/word hash that matches another line/word hash.
Matched section: section that consists completely of linked entries.

Steps

For each transcript file: Create a hash line with each hash representing a line.
For each transcript file: Create a hash line with each hash representing a word.
When creating line hashes ignore white space characters.

Function1 (draftList, finaList)

Function1 is called twice; first with the hash lists of all the lines, then with the hash lists of all the words. When a line is linked it's words are also linked. When we call this function the second time all lines that matched have words that are linked.

Loop (until no more links are made)

if (the first time)
  For each list make a section covering the entire list.
else
  Build section lists for both files by traversing list entries and making a section for each set of adjacent entries that are unlinked and for each set of adjacent entries that are linked to a set of adjacent entries in the other file.
end if

call Function 2

end loop

Function2 (draftList, finaList)

For each unmatched section; for each entry in the section, if the hash appears only once in both lists make a link between them, they match.

Upon linking all unique hashes for the section, also attempt to link the first and last entries of the section. Next, iterate through the section starting from a known linked entry and work both forwards and backwards through the section linking entries that are adjacent, having matching hashes, and are not already linked. Stop at the first unlinkable pair.

The links between the line lists and the word list of the draft and final transcripts are useful in determining whether the final transcript was likely derived from the draft transcript, as previously explained. Further, the links between the word lists created by the operations of Fig. 16 are also used, in accordance with one embodiment of the present invention, to transfer annotations associated with the draft electronic transcript to the corresponding final electronic transcript for use therewith.

After the draft and final transcripts have been compared, the annotations associated with the draft transcript can be relocated and associated with the final transcript, in accordance with one embodiment of the present invention. Fig. 18 illustrates one example of the logical operations for relocating an annotation from a draft transcript into a final transcript for use therewith. At operation [[1]]122, the start and end line positions of the annotation are located within the draft transcript. At operation [[2]]123, the word located at the start position of the annotation in the draft transcript is determined, and the word located at the end position of the annotation within the draft transcript is also located. At operation [[3]]124, the start word is matched to the corresponding word in the final transcript, while the end word is matched to the corresponding word in the final transcript. At operation [[4]]125, the annotations in the final transcript are set based upon the position of the start and end words therein. While this process is described with reference to the word at the start and end of an annotation, other words can also be used as a matter of choice for the particular implementation.

Fig. 19 illustrates an example of the logical operation for transferring annotations associated with the draft electronic transcript to the final electronic transcript, in accordance with one embodiment of the present invention. Examples of the operations of Fig. 19 are illustrated in Figs. 20A-D. The operations of Fig. 19 are described with reference to a single annotation having a start position and an end position associated with a draft transcript. It is understood that a typical draft transcript may have many annotations associated therewith, and accordingly the operations shown in Figs. 19-20 would be preferably be performed on all annotations in the draft transcript.

In Fig. 20A, the example annotation A_D has a start position 126 and end position 127 within the word list of a draft transcript. Fig. 20A also illustrates an example of the links 128 made between entries of the word lists by the operations of Fig. 16. It is noted that the example shown in Figs. 20A-D is a new and different example than the example shown in Figs. 17A-D.

Referring now to Fig. 19, operation [[21]]129 locates the start position of the annotation within the draft transcript. One example of operation [[21]]129 is illustrated in Fig. 20B wherein the start position of the annotation is at the location of the second entry 130 within the word list of the draft transcript. Operation [[22]]131 of Fig. 19 determines if the word entry at the start position of the annotation has a link to an entry in the word list of the final transcript. If so, operation [[22]]131 follows the link to the corresponding word in the final word list. Control is then passed to operation [[28]]132, described below.

If the entry at the start position of the annotation in the draft word list is not linked, then operation [[23]]133 preferably moves up by one or more entries from the start position in the draft word list to find an entry that has a link associated therewith. If no link was found by traversing upwardly until the start of the transcript, operation [[24]]134 repeats operation

[[23]]133 in the opposite direction until a link is found. In this manner, in the event that text was added to the final transcript at the front of an annotation, operation [[23]]133 increases the likelihood that the annotation in the final transcript will include the appropriate text. Operation [[25]]135 counts the number of entries moved until a link was found, as well as the direction in which the movement was made (upward/downward).

Operation [[26]]136 follows the link found in the draft word list to the corresponding entry in the final word list. Operation [[27]]137 travels the reverse distance from the linked entry (or until another link is reached or until the end of the transcript is reached) in the final word list, and control is passed to operation [[28]]132. In this manner, operation [[27]]137 generally accounts for missing text or deleted words of the final transcript.

Operation [[28]]132 sets the current position in the final word list as the start position for the annotation within the final word list. Since each entry within the word list has the page and line number indicating where the word occurred within the final transcript, as well as the start position of the word as measured from the left side of the final transcript and the end position of the word as measured from the right most position of the final transcript, this data is used in locating the starting location of the annotation within the final transcript. Preferably, the page number, line number and starting position of the word in the final word list is used to locate the starting position of the annotation.

Referring to Fig. 20B, one example of the logical operations of Fig. 19 is shown. Beginning with operation [[21]]129, the starting position of the annotation is associated with the second entry 130 in the draft word list. Since this entry 130 does not have a link associated with it as determined by operation [[22]]131, control is passed to operation [[23]]133 which increments the present position upwardly to the first entry 138 in the word list in this example.

Since the top entry 138 in the draft word list has a link 139 associated with it as determined by operation [[24]]134, control is passed to operation [[25]]135 wherein the number of entries moved in order to find a link was one and the direction was upward. Operation [[26]]136 then follows the link 139 associated with the top entry 138 in the draft word list to the associated entry 140 in the final word list, which happens to be the top entry in the final word list in this example. Operation [[27]]137 then travels the reverse distance (in this example, one position downward) in order to reach the proper position with the final word list. Operation [[28]]132 sets this position as the starting point for the annotation in the final word list, preferably using the page number, line number and starting position of the word in the final word list as indicated within the entry to which operation [[28]]132 points.

Referring again to Fig. 19, operation [[28]]132 then passes control to operation [[29]]141 in order to set the end position of the annotation within the final transcript. Operation [[29]]141 repeats operations [[21-28]]129, 131, 133, 135, 136, 137 and 132 beginning at the end position 142 of the annotation in the draft transcript, with these operations generally moving in opposite directions than the directions indicated in operations [[21-27]]129, 131, 133, 134, 135, 136 and 137 to set the end position of the annotation, preferably. In one example the page number, line, and end position of the word as indicated in the entry of the final word list is preferably used to set the end position of the annotation within the final word list.

Fig. 20C illustrates an example of operation [[29]]141 for setting the end position 142 of the annotation in the final transcript. Since operation [[29]]141 repeats steps [[21-28]]129, 131, 133, 134, 135, 136, 137 and 132 as adapted for positioning the end of the annotation within the final transcript, Fig. 20C will be described with reference to steps [[21-28]]129, 131, 133, 134, 135, 136, 137 and 132. Operation [[21]]129 locates the end position 142 of the annotation

[[with]]within the draft transcript, and is shown in Fig. 20C as being the second to last entry within the word list of the draft transcript. Since this entry does not have a link associated therewith as determined by operation [[22]]131, control is passed to operation [[23]]133 which travels downwardly by one entry from the end position to the next entry 143 down in the word list. Operation [[24]]134 determines that a link 145 is found at the present entry 141, and operation [[25]]135 counts the number of entries moved as being one and the direction as being downward when an entry 143 was found having a link 145 associated therewith. Operation [[26]]136 then follows the link 145 from the entry 143 in the draft word list to the corresponding linked entry 146 in the final word list. Operation [[27]]137 then travels a reverse direction in an equal amount from the linked entry 146, and in this example, upwardly by one position. Operation [[28]]132 then sets this position 147 as the end position for the annotation within the final transcript. Preferably, the end position 147 of the annotation within the final transcript is determined using the page number, line number, and end position of the word as indicated within the entry pointed to by operation [[28]]132.

Accordingly, having performed all operations [[21-29]] of Fig. 19 using the starting and ending positions 130,142 of the annotation in the draft transcript, a corresponding annotation in the final transcript is positioned at the proper locations, as illustrated in Fig. 20D, in accordance with one embodiment of the present invention.

The following pseudo-code illustrates one example of the operations illustrated in Fig. 19:

For each annotation associated with the draft transcript; Find the first and last draft lines (hashed entries) that belong to the annotation's start and end points. From these lines find the draft words (hash entries) for the start and end of the annotation. If the annotation does not start or end on a word the closest word is used. In the event of a tie the start position uses the right word and the end position uses the left word.

For each draft word; attempt to obtain the corresponding final word:

The first (simple) case is that the draft word exactly matches a word in the final transcript. (i.e. The draft word is linked to a final word)

The second case is that the word was deleted/modified in the final version. If so get the closest linked draft word next to our draft word. In the case of the starting position word we look to the left first and then to the right, in the case of the end position the search is opposite. From the closest linked draft word get the corresponding final word. From there walk back down the final list the same number of words that we are away (distance from draft to draft word) until either we are at the last word in the transcript or the next word is linked. At that point we have our corresponding final word.

Finally, set the annotations start and end points to the new points calculated above, taking into account if the start and end points have flipped (i.e., if the start is now the end)

Further, in accordance with embodiments of the present invention, a method for analyzing the structure of an electronic transcript is also disclosed herein. This method permits the determination of structural elements of a transcript so that, for example, a hyperlinked table of contents can be displayed to the user within the display pane as shown in Figs. 7-9. In one example, the structural elements include, but are not limited to, pointers to the page where the appearances are listed (i.e., where the transcript identifies the names of the people and parties taking part in the proceeding), pointers to an exhibit index, pointers to the start of the proceedings, pointers to the beginning of a witness' testimony, pointers to the beginning of direct examination of a witness, pointers to the beginning of cross-examination of a witness, etc.

The method described herein can be used for detecting the structural elements of a transcript which is a draft transcript, a final transcript embodied in a file, or a real-time transcript file containing real-time transcript data. The method can also be applied dynamically and on-the-fly to streaming real-time transcript data as it is received by a computing system having the transcript management software operating thereon.

Fig. 21 illustrates the logical operations for analyzing the structural elements of an electronic transcript, in accordance with on embodiment of the present invention. At operation [[1]]150, transcript data is received for processing. As mentioned above, the transcript data can be in the form of a draft of final electronic transcript in a file, or transcript data received from a real-time feed. The operations of Fig. 21 can operate, for example, on transcript files of ASCII format, AMICUS (TM) format, page image format, electronic transcript format as an executable or PTX format, a LIVE NOTE (TM) PTF format, XML format, or the like.

Upon receiving the transcript data, operations [[2-5]]151-154 process the transcript data as will be described in greater detail below. In general, operation [[2]]151 divides the received transcript data into lexical units. Examples of lexical units include simple words, abbreviations that include periods, numerical values with or without commas, monetary values, dates expressed as slash separated numbers, and many others. Operation [[3]]152 divides the received transcript data into actual body text and transcript layout data. The transcript layout data includes, for example, line numbers, page numbers and headers and footers.

Operation [[4]]153 assigns properties to each line of the actual body text. Examples of properties assigned to the lines of actual body text include whether the line is likely to be centered, whether the line contains space-separated text that is likely to be a heading (e.g. I N D E X); whether the line is entirely in uppercase; whether the line possibly indicates that a new speaker has started; whether the current speaker is an attorney, the court, or a witness; the name of the speaker; and many others.

Operation [[5]]154 categorizes each line of the transcript and assigns a structural description thereto. In one example, an engine is used that contains expert knowledge of possible transcript layouts, and uses the information from previous steps to apply heuristics to

each line in turn to determine whether the line of the transcript falls into one of several categories that a user may be interested in. The categories for lines include, for example, headings, start of witness, start of examination (including type of examination and by whom), exhibit reference, and others.

The operations of Fig. 21 will now be described in greater detail. With reference to operation [[2]]151, operation [[2]]151 divides the received transcript data into a sequence of lexical units. Examples of lexical units include simple words, abbreviations that include periods, numerical values with or without commas, monetary values, dates expressed as slash separated numbers, and many others. Preferably, operation [[2]]151 is performed early in the structural analysis process so that the subsequent steps have detailed information about the transcript which can be used without repeated re-classification of the transcript data, which is useful for operations in real-time.

Space characters (' '), tabs ('\t'), line feeds ('\n'), carriage returns ('\r'), end of file markers ('0x1a' and '\0'), and page breaks ('\f') are the only characters guaranteed to separate lexical units or tokens (the values given in parentheses are the C language value of these characters). Many other characters may separate or may be part of a token. For example, a period "." may appear as part of an abbreviation (such as "etc.") or may end a sentence.

Preferably, the lexical units can include a type field, a subtype field, and a content field. Further, each lexical unit has a trailer which points to the next lexical unit in the sequence. The content field includes the content from the transcript data. The type field and the sub-type field describe the type of lexical unit, and preferably can include the following:

Integer: A whole number. Such as 17 or 37,000.

Real: A number followed by a period and additional numbers. Such as 2.45 or 45,567.45

Phone Number: A number containing hyphens. E.g. 345-234-2345

Ordinal: One or more digits followed by either st, nd, th, rd, or d. E.g. 31st. 193rd.

Time: A time in the format HH:MM:SS followed by either AM or PM. For example 4:27PM, or 09:10:55.

Date: A slash separated date. For example 5/1999 or 04/19/2000.

Currency: An Integer or Real immediately preceded by a dollar sign. E.g. $17 or $1,234.45

HashNumber: An Integer immediately preceded by a pound sign. E.g. #123

Abbreviation: Any sequence of letters and periods beginning with a letter that contains at least four characters total. Preferably, most end of sentence words are treated as abbreviations.

Initial: A single letter followed by a period.

Word: Any sequence of letters that may contain apostrophes, or end in an 's' followed by an apostrophe.

AlphaNum: Any sequence of letters, numerals, apostrophes and hyphens.

Punctuation: Classified into several sub-types, they are:

Phrase separators: Comma, semi-colon or colon.

End of sentence: Period, exclamation mark, or question mark.

Open group: Left parenthesis or left square bracket.

Close group: Right parenthesis or right square bracket.

Quote: Double quote, Single quote or Back-quote.

Pause: Slash (forward or backward), isolated hyphen, or three periods (an ellipsis).

Formatting: Two sub-types:

End-of-line: Any single carriage return or line feed, or pair thereof.

End-of-file: The 'Ox l a' or '\0' characters.

Whitespace: Each tab character is converted to five spaces that are also combined with any adjacent spaces. Any sequence of spaces is treated as a single token.

Misc: Any sequence of characters that is not classified into one of the above categories is treated as a miscellaneous token. This includes any sequence wholly containing the following characters _@#$%^&*+-=}{ | ~.

The process of breaking the received transcript data down into lexical units preferably takes place on a line-by-line basis. For any given chunk of raw transcript text, the result of this process yields a structure similar to that shown in Figs. 22A-B. Fig. 22A shows an example of original transcript data 155 having a page break character 156, a page number 157, a text section 158 identifying the "direct examination", text of the questions and answers transcribed 159, and line numbers 160 corresponding thereto.

Fig. 22B shows an example of how the original transcript data from Fig. 22A is processed into various lexical units. As shown in Fig. 22B, preferably a line index 161 is maintained, and for each line index, a type field 162, a subtype field 163, and a content field 164 is maintained for arranging the transcript data into its various lexical units, as described above.

This resulting lexical units created by operation [[2]]151 are used during later operations within the process, such as operations [[3-5]]152-154 of Fig. 21

With reference to operation [[3]]152 of Fig. 21, operation [[3]]152 divides the received transcript data into actual body text and transcript layout data. The transcript layout data includes, for example, line numbers, page numbers and headers and footers.

Operation [[3]]152 operates on ASCII transcripts in page image format and transcripts in other formats.

The layout of a typical ASCII transcript page can be broken down into discrete sections, as follows: 1) header and footer [optional]; 2) left margin [may be zero]; 3) line numbers [optional]; 4) text leader; 5) line text; and 6) page break to separate from next page [optional].

Fig. 23 shows an example of the start of a typical page image ASCII transcript page. In Fig. 23, the header 165, line text 166, line numbers 167 and page number 168 are labeled. The left margin is the space to the left of the line numbers, and the text leader is the space between the line numbers 167 and the line text 166.

The process of taking a page image ASCII file and normalizing it into these component elements is adapted to handle many different transcript formats. Many elements are optional, or due to limitations in the software that generates the transcript file, are simply not present. This normalization of the transcript text works with the pre-parsed transcript (i.e. after it has been broken down into lexical units by operation [[2]]151). This ensures that the normalization occurs as quickly as possible.

The normalization phase of operation [[3]]152 preferably characterizes each line as:

Whether it is line numbered.

Whether it contains a page number.

Whether it is a header or footer line.

The index of the first lexical unit that is actual transcript text (if any). [Note that header and footer lines are preferably not treated as part of the transcript text.]

The total number of lexical units it contains.

The width of the actual transcript text, in characters.

The width of the leader text for the line, in characters.

Whether the line is in between numbered lines (title and index pages often have unnumbered lines between the numbered lines that are part of the transcript text).

The width of the leader on each page. Note that some lines may be indented further in addition to the leader.

With respect to operation [[4]]153 of Fig. 21, operation [[4]]153 assigns properties to each line of the actual body text. Examples of properties assigned to the lines of actual body text include whether the line is likely to be centered, whether the line contains space-separated text that is likely to be a heading (e.g. I N D E X); whether the line is entirely in uppercase; whether the line possibly indicates that a new speaker has started; whether the current speaker is an attorney, the court, or a witness; the name of the speaker; and many others. Various properties are assigned to the lines, such as:

Whether it is written as expanded text (e.g. E X H I B I T I N D E X)

The number of terms of more than a single space in length.

Whether the line is all upper case.

Whether the line starts with an all upper case word.

The position of the first word on the line that is not all upper case.

Whether the line begins with an open parenthesis.

Whether the line ends with a close parenthesis.

Whether the line ends with a colon.

Whether the line ends in an integer.

Whether the line ends in a period.

The position of the word "BY" word in the line.

The position of the first colon in the line.

The position of the first comma in the line.

The position of the first close paren in the line.

The position of the first "EXAMINATION" word in the line.

The position of first "DEPOSITION" word in the line.

Whether the line is centered (fuzzy calculation).

Whether the line is probably inside a parenthetical (fuzzy calculation).

Whether the line consists of spoken text.

The type of speech this line is part of: None, Court, Answer, Question, Attorney, or Witness.

A determination is made whether the line is expanded. By looping through the terms in a line, it can be seen whether the line consists of single letters that are separated by spaces (and words that are separated by three or more spaces). If it does, then the actual non-expanded text of the line is calculated and carried around with the line data, and preferably a "Line is expanded" flag is set.

A determination is made whether the line is centered. This is an indication that the line is probably centered, and is treated as such when computing the actual structure. Whether the line is to be considered centered is computed as follows:

1. Compute the amount of space on the left of the text, by taking the first whitespace token and subtracting the width (in characters) of the left leader.

2. Do a check on the left indent. If the left indent is less than 6 or more than 33 (for example), then the line is never be marked as centered.

3. Compute the amount of space on the right of the text, by subtracting the width of the line (including the space on the left) from the actual width, in characters, of the transcript

page. The page width is assumed to be the maximum width of the transcript text lines on the current page (i.e. not including margin, line number, or text leader).

4. Compute the absolute value of the difference between the amount of space on the right, and the amount of space on the left.

5. If this difference is less than one-third of the size of the left indent, then set a Centered flag to True. Otherwise the centered flag is set to False.

A determination is made whether the line is within a parenthetical. A parenthetical is an aside inserted by a court reporter into the transcript text when some non spoken action takes place, or when a summary of an action is required. For example, "(The jury entered the court-room)". These parentheticals may span lines, and text within them is not treated as spoken text, or as a potential heading.

To determine if a line is within a parenthetical:

1. If the line begins and ends with a parenthesis then it is marked as being in a parenthetical.

2. If we started a line with an open parenthesis less than 7 lines ago, for which there has been no matching close parenthesis, then mark this line as within a parenthetical.

Preferably, only 7 full lines are permitted for a parenthetical to prevent unmatching parentheses from causing an entire transcript to be ignored for structure.

With respect to operation [[5]]<u>154</u> of Fig. 21, operation [[5]]<u>154</u> categorizes each line of the transcript and assigns a structural description thereto. In one example, an engine is used that contains expert knowledge of possible transcript layouts, and uses the information from previous steps to apply heuristics to each line in turn to determine whether the line of the transcript falls into one of several categories that a user may be interested in. The categories for lines include,

for example, headings, start of witness, start of examination (including type of examination and by whom), exhibit reference, and others.

As a result of the line properties gathered during the previous phase, the structure of the transcript can be extracted. The following four structural entries are detected during operation [[5]]<u>154</u>:

1. Speaker names.
2. Headings (e.g. Exhibit Index).
3. Start of witness testimony.
4. Witness sections (e.g. Cross-examination).

The additional properties that are gathered to identify this structure are as follows:

Whether the line contains a speaker name.

The name of the speaker name this line contains (if any e.g. Mr. Smith).

Whether the line is a heading

The name of the heading this line contains (if any, e.g. Exhibit Index).

Whether this line contains a witness section term (i.e. a term that indicates that a new examination, cross-examination etc. is starting).

Whether this line begins a new witness section.

The name of the witness section. (E.g. Direct Examination)

The name of the person examining the witness (e.g. Mr. Smith)

Whether this line contains a witness name term (i.e. a term that indicates a new witness is or will be starting).

Whether this line possibly contains a witness name.

The name of the witness.

Whether the possible witness name has been confirmed as a witness name.

The process of determining these properties is as follows:

Determine if a line matches the list of "inclusions" for the structure entry type. E.g. If it is centered, expanded or all uppercase it is further processed to see if it is actually a heading.

Filter out those lines that match any set of properties in the exclusion list for structure entries of the matching type. E.g., if it does not contain one of the words that are required in headings, then it is not treated as a heading.

Developing the list of inclusions and exclusions for each structure entry type is the main detail in this section. Doing so relies on iterative testing of many sample transcripts. In each iteration, an expert identifies the reasons for noise matches, and adds "exclusions" to each structure entry type within the engine to prevent those matches from occurring again. In doing so, it is important to be careful to not exclude legitimate matches, and additional "inclusion" rules may be needed to ensure that appropriate lines are passed through the exclusion filter.

An example method of determining whether a line is a new witness section (Examination, Cross-Examination etc.) is shown in Fig. 24. Similar methods can be employed for each type of structure entry. Referring to Fig. 24, at operation [[1]]175 the line is examined for example to determine whether it contains the word "EXAMINATION." If not, then control is passed to operation 2 and the line is not a change of witness section.

If the line does contain "EXAMINATION" as determined by operation [[1]]175, then decision operation [[3]]176 determines if the line contains the word "BY." If not, control is passed to operation [[4]]177 which tests whether the line contains all caps and centered text but does not end in a period. If no, decision operation [[5]]178 determines if the line is expanded, and if not, decision operation [[6]]179 determines whether the line starts capitalized only, but

"EXAMINATION" is in all caps. If not, then control is passed to operation [[7]]180 as this is not a witness section. If either decision operations ~~3, 4, 5 or 6,~~ 176, 177, 178 or 179 result in a yes, then control is passed to operation [[8]]181 which determines whether the line is on the same page as a heading such as "Index," "Contents," or "Proceedings." If so, control is passed to operation [[7]]180, as this is not a witness section. Otherwise, decision operation [[8]]181 passes control to operation [[9]]182 as the line is a new witness section.

From the operations of Figs. 21-24, the analysis of the structural components of a transcript can be used for a variety of purposes, including for example, creating a table of contents having hyper-links thereto showing the user the desired structural sections, as discussed above.

Embodiments of the present invention can be embodied in a computer program product. It will be understood that the computer program product of the present invention preferably is created in a computer usable medium, having computer readable code embodied therein. The computer usable medium preferably contains a number of computer readable program code devices configured to cause a computer to affect the various functions required to carry out the invention, as herein described.

The embodiments of the invention described herein are implemented as logical operations in a computing system. The logical operations of the present invention are implemented (1) as a sequence of computing implemented steps running on the computing system and (2) as interconnected machine modules within the computing system. The implementation is a matter of choice dependent on the performance requirements of the computing system implementing the invention. Accordingly, the logical operations making up

the embodiments of the invention described herein are referred to variously as operations, steps, or modules.

While the methods disclosed herein has been described and shown with reference to particular steps or operations performed in a particular order, it will be understood that these steps or operations may be combined, sub-divided, or re-ordered to form an equivalent method without departing from the teachings of the present invention. Accordingly, unless specifically indicated herein, the order and grouping of the steps is not a limitation of the present invention.

The foregoing embodiments and examples are to be considered illustrative, rather than restrictive of the invention, and those modifications, which come within the meaning and range of equivalence of the claims, are to be included therein. While the invention has been particularly shown and described with reference to a preferred embodiment thereof, it will be understood by those skilled in the art that various other changes in the form and details may be made without departing from the spirit and scope of the invention.